In the Claims

Claims 8-13 are canceled without prejudice.

Claims 14 and 15 are amended.

Claims 1-4, 6, 7, 14, 15 and 18-22 remain in the application and are listed below.

1. (Previously Presented) A method for generating a delta between a first program binary and a second program binary, the method comprising the steps of:

obtaining a first control flow graph (CFG) representation of the first binary and obtaining a second CFG representation of the second binary;

comparing the first and second CFG representations to identify blocks (nominally matched blocks) that match in the first and second CFG representations, thereby identifying blocks (nominally unmatched blocks) in the second CFG representation that do not match in the first CFG representation, the comparing being based upon content of blocks being compared and augmented local neighborhoods of blocks surrounding blocks being compared, wherein a local neighborhood of a particular block consists of blocks neighboring that block in a CFG representation, but less than all the blocks in that CFG representation, and an augmented local neighborhood of that particular block consists that block's local neighborhood plus a random sampling of blocks from a substantially larger neighborhood of blocks surrounding that block, an augmented local neighborhood in a CFG representation consisting of less than all the blocks in that CFG representation;

determining edit-operations that merges the unmatched blocks into the first CFG representation so that first CFG representation is substantially identical to the second CFG representation;

producing a delta comprising the unmatched blocks and the edit-operations.

2.   (Original)   A method for transmitting a delta comprising:

a method for generating a delta as recited in claim 1;

transmitting the delta over a network.

3.   (Original)   A method for patching a copy of the first program binary, the method comprising:

a method for generating a delta as recited in claim 1;

patching the copy of the first program binary so that the copy is substantially identical to the second program binary, wherein the delta guides such patching.

4.   (Previously Presented)   A method as recited in claim 1, wherein a local neighborhood of a particular block consists of those blocks immediately adjacent that block.

5.   (Canceled)

6. (Original) A computer-readable medium having embodied thereon a data structure, comprising a delta generated in accordance with the steps recited in claim 1.

7. (Original) A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 1.

8. (Canceled).

9. (Canceled).

10. (Canceled).

11. (Canceled).

12. (Canceled).

13. (Canceled).

14.   (Currently Amended)   ~~A method as recited in claim 13,~~ A method for matching of blocks in a procedure of a first control flow graph (CFG) representation of a portion of a first program between an ostensibly matching procedure of a second CFG representation of a portion of a second program, the method comprising:

matching blocks between the first and second CFG representations based upon the content of the blocks;

computing successively smaller neighborhoods of each block in the first and second CFG representations via breadth first traversals;

for each neighborhood computed in the computing step, forming a "d-label" for each block in a neighborhood based upon labels of the blocks within the neighborhood;

attempting to match blocks between first and second CFG representations by comparing the d-labels of the blocks, wherein at least one neighborhood computed in the computing steps is augmented with a random sampling of blocks in the complete representation of the neighborhood.

15.   (Currently Amended)   A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim [[13]] 14.

16.   (Not Entered)

17.   (Not Entered)

18.   (Previously Presented)   A patch data structure generated in accordance with the following acts:

providing a server computer in a communications with a communications network;

receiving input from a client computer by way of the communications network, the input providing a parameter indicative of a request for upgrading a copy of a first program binary to a match a second program binary;

retrieving a delta between a first program binary and the second program binary, wherein computing such delta comprises the steps of:

a) obtaining a first control flow graph (CFG) representation of the first binary and obtaining a second CFG representation of the second binary;

b) comparing the first and second CFG representations to identify blocks (nominally matched blocks) that match in the first and second CFG representations, thereby identifying blocks (nominally unmatched blocks) in the second CFG representation that do not match in the first CFG representation, the comparing being based upon content of blocks being compared and augmented local neighborhoods of blocks surrounding blocks being compared, wherein a local neighborhood of a particular block consists of blocks neighboring that block in a CFG representation, but less than all the blocks in that CFG representation, and an augmented local neighborhood of that particular block consists that block's local neighborhood plus a random sampling of blocks from a substantially larger neighborhood of blocks surrounding that block, an augmented

local neighborhood in a CFG representation consisting of less than all the blocks in that CFG representation;

c) determining edit-operations that merges the unmatched blocks into the first CFG representation so that first CFG representation is substantially identical to the second CFG representation;

d) producing a delta comprising the unmatched blocks and the edit-operations;

generating the patch data structure as a function of the delta.

19. (Original)   A method for transmitting a patch data structure comprising transmitting a patch data structure as recited in claim 18 over a communications network.

20. (Original)   A method for patching a copy of the first program binary at a client computer, the method comprising patching the copy of the first program binary so that the copy is substantially identical to the second program binary, wherein a delta in a patch data structure as recited in claim 18 guides such patching.

21.   (Previously Presented)   A delta-generator system, comprising:

a comparator that is configured to compare a first control flow graph (CFG) representation of a first program binary and a second CFG representation of the second program binary for identifying blocks (nominally matched blocks) that match in the first and second CFG representations, thereby identifying blocks (nominally unmatched blocks) in the second CFG representation that do not match in the first CFG representation, the comparing being based upon content of blocks being compared and augmented local neighborhoods of blocks surrounding blocks being compared, wherein a local neighborhood of a particular block consists of blocks neighboring that block in a CFG representation, but less than all the blocks in that CFG representation, and an augmented local neighborhood of that particular block consists that block's local neighborhood plus a random sampling of blocks from a substantially larger neighborhood of blocks surrounding that block, an augmented local neighborhood in a CFG representation consisting of less than all the blocks in that CFG representation;

an edit-op determiner configured to determine the edit-operations that merges the unmatched blocks into the first CFG representation so that first CFG representation is substantially identical to the second CFG representation;

an output sub-system that is configured to produce a delta comprising the unmatched blocks and the edit-operations.

22.   (Original)   A computer-readable medium having embodied thereon a data structure comprising a delta produced by the system as recited in claim 21.